REMARKS

In the non-final Office Action, the Examiner rejected claim 16 under 35 U.S.C. § 102(e) as anticipated by Adya et al. (U.S. Patent Application Publication No. 2005/0102268); rejected claims 1-10, 12-15, and 17 under 35 U.S.C. § 103(a) as unpatentable over Adya et al. in view of McClaughry et al. (U.S. Patent No. 5,933,825); and rejected claim 11 under 35 U.S.C. § 103(a) as unpatentable over Adya et al. in view of McClaughry et al. and Robinson (U.S. Patent No. 4,709,326).

By this Amendment, Applicants amend claims 1, 4, 9-11, and 13-17 to improve form. Applicants respectfully traverse the Examiner's rejections under 35 U.S.C. §§ 102 and 103. Claims 1-17 remain pending.

REJECTION UNDER 35 U.S.C. § 102 BASED ON ADYA ET AL.

In paragraph 3 of the Office Action, the Examiner rejected claim 16 under 35 U.S.C. § 102(e) as allegedly anticipated by Adya et al. Applicants respectfully traverse the rejection.

A proper rejection under 35 U.S.C. § 102 requires that a single reference teach every aspect of the claimed invention either expressly or impliedly. Any feature not directly taught must be inherently present. In other words, the identical invention must be shown in as complete detail as contained in the claim. See M.P.E.P. § 2131. Adya et al. does not disclose or suggest the combination of features recited in amended claim 16.

For example, amended independent claim 16 is directed to a method for performing first and second operations within a same directory. The method comprises obtaining a first lock on a sub-directory or file name within the directory by the first operation; obtaining a second lock on a sub-directory or file name within the directory by the second operation; determining whether the first and second locks conflict; concurrently performing the first and second operations when the first and second locks do not conflict, the first and second locks being read-write locks; and serializing performance of the first and second operations when the first and second locks conflict.

Adya et al. does not disclose or suggest the combination of features recited in amended claim 16. For example, Adya et al. does not disclose or suggest serializing performance of first and second operations when a first lock obtained by the first operation conflicts with a second lock obtained by the second operation. Instead, Adya et al. discloses denying a lock request when a conflict exists (para. 0135).

When rejecting a similar feature recited in claim 9, the Examiner alleged that Adya et al. discloses serializing first, second, and third locks and cited paragraphs 0123, 0131-0133, and 0137 of Adya et al. for support (Office Action, page 5). Applicants respectfully submit that nowhere in these sections, or elsewhere, does Adya et al. disclose or suggest serializing performance of first and second operations when a first lock obtained by the first operation conflicts with a second lock obtained by the second operation, as required by claim 16.

At paragraph 0123, Adya et al. discloses:

> When an application desires to open an object, the directory group performs two checks: (1) are the modes the application is asking for going to conflict with another application that has already opened the object; and (2) are the operations that the application is willing to share the object for going to conflict with what another application has already opened the object for and indicated it is willing to share the object for. Six of the ten locks are directed to supporting this checking: Open Read, Open Write, Open Delete, Open Not Shared Read, Open Not Shared Write, and Open Not Shared Delete. These locks are used to grant an application the ability to open an object, but do not necessarily guarantee that the data for the object can be obtained (the Read lock or Write lock (depending on the type of operation the application desires to perform) is obtained to access the data). Open Read Lock. The Open Read lock is requested by an application to allow the application to open the associated object for reading.

In this section, Adya et al. discloses that when an application requests to open an object, it is determined whether the modes that the application is requesting conflict with another application that has already opened the object and whether the operations for which the application is willing to share conflict with another application that has already opened the object. Nowhere in this section, or elsewhere, does Adya et al. disclose or suggest serializing performance of first and second operations when a first lock obtained by the first operation conflicts with a second lock obtained by the second operation, as required by claim 16.

At paragraphs 0131-0133, Adya et al. discloses:

> Exclusive Lock. The Exclusive lock is requested by an application to obtain all of the previously discussed nine locks, including an Insert lock on each possible name that could exist (but does not already exist) in the directory. An Exclusive lock on a directory does not imply Exclusive locks on the files or subdirectories in the directory, but rather only on the directory's namespace. The Exclusive lock conflicts with each of the previously discussed nine locks.
>
> Various conflicts exist between the various different locks. Table I is a conflict matrix illustrating the conflicts between locks in one exemplary implementation. The following abbreviations are used in Table I: Ins (Insert), Excl (Exclusive), O-R (Open Read), O-W (Open Write), O-D (Open Delete), O-!R (Open Not Shared Read), O-!W (Open Not Shared Write), and O-!D (Open Not Shared Delete). An "X" in a cell of Table I indicates a conflict between the corresponding two locks-for example, Open Read conflicts with Open Not Shared Read but does not conflict with Open Not Shared Write.
>
> FIG. 9 is a flowchart illustrating an exemplary process for determining whether to allow a particular object to be opened. The process of FIG. 9 is implemented by the directory group responsible for managing the particular object. In the process of FIG. 9, it is assumed that the client requesting to open the particular object does not already have the necessary lock(s) to open the object as desired. Initially, a request to access an object with particular locks identified is received (act 902). A check is made by the directory group as to whether the modes implied by the selected locks conflict with locks that have been granted to a different client (act 904). For example, if the request is a request to open an object for reading, but another application has already opened the object with the Not Shared Read lock, then the mode (open read) implied by the selected lock conflicts with another application that has already opened the object. Because the directory group knows only if it has issued a conflicting lock to a client, but not whether the client is currently using the lock to allow an application access to an object, in some cases making the check in act 904 requires asking a client that currently holds a lock is willing to give it up.

In this section, Adya et al. discloses information regarding an exclusive lock, information regarding the various conflicts that exist between the various different locks, and that a check is made whether the modes implied by locks requested by a client conflict with locks that have been granted to a different client. Nowhere in this section, or elsewhere, does Adya et al. disclose or suggest serializing performance of first and second operations when a first lock obtained by the first operation conflicts with a second lock obtained by the second operation, as required by claim 16.

At paragraph 0137, Adya et al. discloses:

> If a client's lock request conflicts with an existing lock granted to another client, the directory group may attempt to downgrade the earlier-issued lock to one that will not conflict with the new request at act 910 (e.g., rather than denying the request in act 914). Since lock upgrades result in clients holding locks that they did not request, lock downgrades typically have a non-trivial likelihood of success. If the lock recall fails, then the request is denied.

In this section, Adya et al. discloses that if a client's lock request conflicts with an existing lock granted to another client, the earlier issued lock may be downgraded so that the lock request will no longer conflict. If this fails, Adya et al. discloses that the lock request is denied. Nowhere in this section, or elsewhere, does Adya et al. disclose or suggest serializing performance of first and second operations when a first lock obtained by the first operation conflicts with a second lock obtained by the second operation, as required by claim 16. In fact, Adya et al. actually teaches away from serializing by disclosing that a lock request is denied when a conflict exists.

For at least these reasons, Applicants submit that claim 16 is not anticipated by Adya et al.

REJECTION UNDER 35 U.S.C. § 103 BASED ON ADYA ET AL. AND McCLAUGHRY ET AL.

In paragraph 4 of the Office Action, the Examiner rejected claims 1-10, 12-15, and 17 under 35 U.S.C. § 103(a) as allegedly unpatentable over Adya et al. in view of McClaughry et al. Applicants respectfully traverse the rejection.

Amended independent claim 1, for example, is directed to a method for performing operations within a file system in which directories and files are organized as nodes in a namespace tree. The method comprises associating a read-write lock with each of the nodes in the namespace tree; acquiring a first lock on a name of one or more directories involved in a first operation; acquiring a second lock on an entire pathname involved in the first operation; determining whether the first lock or the second lock conflicts with third locks acquired by a second operation; performing the first operation when the first lock or the second lock does not conflict with the third locks, where the first, second, and third locks are read-write locks; and serializing performance of the first and second operations when the first lock or the second lock conflicts with the third locks.

Neither Adya et al. nor McClaughry et al., whether taken alone or in any reasonable combination, discloses or suggests the combination of features recited in amended claim 1. For example, neither Adya et al. nor McClaughry et al. discloses or suggests serializing performance of first and second operations when a first lock or a second lock acquired by the first operation conflicts with third locks acquired by the second operation.

As explained above with regard to claim 16, Adya et al. teaches away from this feature. McClaughry et al. does not cure this deficiency in the disclosure of Adya et al. For example, McClaughry et al. discloses denying a request for a lock when a conflict exists and notifying the client (col. 3, lines 14-17; col. 5, lines 53-58; and col. 6, lines 42-46). Therefore, neither Adya et al. nor McClaughry et al., whether taken alone or in any reasonable combination, discloses or suggests serializing performance of first and second operations when a first lock or a second lock acquired by the first operation conflicts with third locks acquired by the second operation, as required by claim 1.

For at least these reasons, Applicants submit that claim 1 is patentable over Adya et al. and McClaughry et al., whether taken alone or in any reasonable combination. Claims 2-10 and 12 depend from claim 1 and are, therefore, patentable over Adya et al. and McClaughry et al. for at least the reasons given with regard to claim 1. Claims 2-10 and 12 are also patentable over Adya et al. and McClaughry et al. for reasons of their own.

For example, claim 8 recites using a lazily allocated data structure that maps pathnames to locks to determine whether the first lock or the second lock conflicts with the third locks. Neither Adya et al. nor McClaughry et al., whether taken alone or in any reasonable combination, discloses or suggests the feature recited in claim 8.

The Examiner alleged that Adya et al. discloses the feature of claim 8 and cited paragraphs 0046, 0102-0106, 0115, 0123, and 0131-0133 of Adya et al. for support (Office Action, page 5). Applicants respectfully disagree.

At paragraph 0046, Adya et al. discloses:

> Generally, according to exclusive encryption, a plaintext name (the file or directory name within the directory entry) is mapped to a new name. The mapped name is optionally decasified into a decasified (case-insensitive) name and corresponding case information, allowing duplicate name detection to be case-insensitive. The mapped (and optionally decasified) name is then encoded and encrypted. This encrypted name (and optionally accompanying case information) are forwarded to the directory group that is responsible for managing the directory entry (e.g., based on pathname, as discussed in more detail below).

In this section, Adya et al. discloses that a file or directory name is mapped to a new name that is optionally made case-insensitive. Nowhere in this section, or elsewhere, does Adya et al. disclose or remotely suggest using a lazily allocated data structure that maps pathnames to locks to determine whether the first lock or the second lock conflicts with the third locks, as required by claim 8.

At paragraphs 0102-0106, Adya et al. discloses that each computer in the file system can maintain a local cache that maps a subset of the pathnames in the name space to the directory group that manages that pathname. Nowhere in this section, or elsewhere, does Adya et al. disclose or remotely suggest using a lazily allocated data structure that maps pathnames to locks to determine whether the first lock or the second lock conflicts with the third locks, as required by claim 8.

At paragraph 0115, Adya et al. discloses:

> FIG. 8 is a flowchart illustrating an exemplary process for storing a file in a serverless distributed file system. Initially, a new file storage request is received at a client computing device (act 802). The client encrypts the file and the file name and generates the file contents hash (act 804). The client sends the encrypted file name and file contents hash to the appropriate Byzantine.-fault-tolerant directory group along with a request to create a directory entry (act 806). The directory group validates the request (act 808), such as by verifying that the file name does not conflict with an existing name and that the client has permission to do what it is requesting to do. If the request is not validated then the request fails (act 810). However, if the request is validated, then the directory group generates a directory entry for the new file (act 812). The directory group also determines the replica set for the new file and adds the replica set to the newly generated directory entry (act 814). Replicas of the file are also generated (act 816), and saved to multiple computers in the file system (act 818).

In this section, Adya et al. discloses that a client encrypts a file and a file name and generates a file contents hash and sends this information to a directory group that validates the request. Nowhere in this section, or elsewhere, does Adya et al. disclose or remotely suggest using a lazily allocated data structure that maps pathnames to locks to determine whether the first lock or the second lock conflicts with the third locks, as required by claim 8.

Paragraph 0123 of Adya et al. has been reproduced above. In paragraph 0123, Adya et al. discloses that when an application requests to open an object, it is determined whether the modes that the application is requesting conflict with another application that has already opened the object and whether the operations for which the application is willing to share conflict with another application that has already opened the object. Nowhere in this section, or elsewhere, does Adya et al. disclose or remotely suggest using a lazily allocated data structure that maps pathnames to locks to determine whether the first lock or the second lock conflicts with the third locks, as required by claim 8.

Paragraphs 0131-0133 of Adya et al. have been reproduced above. In paragraphs 0131-0133, Adya et al. discloses information regarding an exclusive lock, information regarding the various conflicts that exist between the various different locks, and that a check is made whether the modes implied by locks requested by a client conflict with locks that have been granted to a different client. Nowhere in this section, or elsewhere, does Adya et al. disclose or remotely suggest using a lazily allocated data structure that maps pathnames to locks to determine whether the first lock or the second lock conflicts with the third locks, as required by claim 8.

McClaughry et al. does not cure the deficiencies in the disclosure of Adya et al.

For at least these additional reasons, Applicants submit that claim 8 is patentable over Adya et al. and McClaughry et al., whether taken alone or in any reasonable combination.

Claim 9 recites serializing the first, second, and third locks when the first lock or the second lock conflicts with the third locks. Neither Adya et al. nor McClaughry et al., whether taken alone or in any reasonable combination, discloses or suggests the feature recited in claim 9.

The Examiner alleged that Adya et al. discloses the feature of claim 9 and cited paragraphs 0123, 0131-0133, and 0137 of Adya et al. for support (Office Action, page 5). Applicants respectfully disagree.

Paragraph 0123 of Adya et al. has been reproduced above. In paragraph 0123, Adya et al. discloses that when an application requests to open an object, it is determined whether the modes that the application is requesting conflict with another application that has already opened the object and whether the operations for which the application is willing to share conflict with another application that has already opened the object. Nowhere in this section, or elsewhere, does Adya et al. disclose or suggest serializing first, second, and third locks when the first lock or the second lock conflicts with the third locks, as required by claim 9.

Paragraphs 0131-0133 of Adya et al. have been reproduced above. In paragraphs 0131-0133, Adya et al. discloses information regarding an exclusive lock, information regarding the various conflicts that exist between the various different locks, and that a check is made whether the modes implied by locks requested by a client conflict with locks that have been granted to a different client. Nowhere in this section, or elsewhere, does Adya et al. disclose or suggest serializing first, second, and third locks when the first lock or the second lock conflicts with the third locks, as required by claim 9.

Paragraph 0137 of Adya et al. has been reproduced above. In paragraph 0137, Adya et al. discloses that if a client's lock request conflicts with an existing lock granted to another client, the earlier issued lock may be downgraded so that the lock request will no longer conflict. If this fails, Adya et al. discloses that the lock request is denied. Nowhere in this section, or elsewhere, does Adya et al. disclose or suggest serializing first, second, and third locks when the first lock or the second lock conflicts with the third locks, as required by claim 9.

McClaughry et al. does not cure the deficiencies in the disclosure of Adya et al.

For at least these additional reasons, Applicants submit that claim 9 is patentable over Adya et al. and McClaughry et al., whether taken alone or in any reasonable combination. Claim 10 depends from claim 9 and is, therefore, also patentable over Adya et al. and McClaughry et al. for at least the reasons given with regard to claim 9.

Amended independent claims 13-15 and 17 recite features similar to, but possibly different in scope from, features recited in claim 1. Claims 13-15 and 17 are, therefore, patentable over Adya et al. and McClaughry et al., whether taken alone or in any reasonable combination, for at least reasons similar to reasons given with regard to claim 1.

REJECTION UNDER 35 U.S.C. § 103 BASED ON ADYA ET AL., McCLAUGHRY ET AL., AND ROBINSON

In paragraph 5 of the Office Action, the Examiner rejected claim 11 under 35 U.S.C. § 103(a) as allegedly unpatentable over Adya et al. in view of McClaughry et al. and Robinson. Applicants respectfully traverse the rejection.

Claim 11 depends from claim 9. Without acquiescing in the Examiner's rejection with regard to claim 11, Applicants respectfully submit that the disclosure of Robinson does not cure the deficiencies in the disclosure of Adya et al. and McClaughry et al. identified above with regard to claim 9. Therefore, claim 11 is patentable over Adya et al., McClaughry et al., and Robinson for at least the reasons given with regard to claim 9.

For at least these reasons, Applicants submit that claim 11 is patentable over Adya et al., McClaughry et al., and Robinson, whether taken alone or in any reasonable combination.

CONCLUSION

In view of the foregoing amendments and remarks, Applicants respectfully request the Examiner's reconsideration of the application and the timely allowance of pending claims 1-17.

If the Examiner does not believe that all pending claims are now in condition for allowance, the Examiner is urged to contact the undersigned to expedite prosecution of this application.

To the extent necessary, a petition for an extension of time under 37 C.F.R. § 1.136 is hereby made. Please charge any shortage in fees due in connection with the filing of this paper, including extension of time fees, to Deposit Account No. 50-1070 and please credit any excess fees to such deposit account.

Respectfully submitted,

HARRITY SNYDER, L.L.P.

By: /Paul A. Harrity/
Paul A. Harrity
Reg. No. 39,574

Date: May 2, 2006

11350 Random Hills Road
Suite 600
Fairfax, Virginia 22030
(571) 432-0800